Electronic Version
Stylesheet Version v1.1.1

## Description

# [System and Method to distribute reasoning and pattern matching in forward and backward chaining rule engines]

### BACKGROUND OF INVENTION

[0001] Rules engine technology is the product of research in Artificial Intelligence and intelligent systems. Throughout the 1970's, 80's and 90's, researchers have solved some complex computing challenges. One of the most efficient and well tested algorithms is RETE. It was originally described by Charles Forgy.

[0002] REFERENCE:1. Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem* by Charles L. Forgy Department of Computer Science. Carnegie-Mellon University.

### DETAILED DESCRIPTION

[0003] Distributed reasoning, unlike collaborative agents provides methods by which multiple rule engines reason over large datasets in real-time. Whereas collaborative agents use agents to reason over discrete problems, it cannot reason over large sets of data. Furthermore, collaborative agents" techniques require the engine to load all necessary data within the same engine. For small datasets, these techniques prove to be powerful, but they do not scale for large datasets.

[0004] A forward chaining rule engine utilizing RETE algorithm, can reason over large datasets when nodes are distributed to other systems. This allows the system to match on a specific instance of a class (also called business object) without requiring the object instance reside in the same working memory and be locally available. It is important to note the engine will reason over shared data resident in other engines using data steams. This reduces the memory requirements and improves efficiency. It also enables the engines to share all or part of their reasoning network and distribute the process. Distributing nodes of a set of rules across multiple systems allows each engine to perform pattern matching on a single object instance and route the results to the originating system. Unlike load balancing techniques, a true distributed reasoning system does not require all systems of a cluster to deploy identical sets of rules, which creates redundant rules within the environment and increases maintenance costs. In a distributed reasoning/distributed pattern matching system, each system deploys a different set of rules (also called module or rule set), based on its own needs and configuration requirements. At runtime, the rule engines monitor resource utilization and distribute nodes dynamically and on demand.

[0005] Prior techniques relied on load balancing techniques to prioritize and categorize atomic processes. This approach requires every system to have all required data locally, leading to multiple data caches. In a production environment with large datasets, each system cannot load all required data and data synchronization becomes a potential problem.